Automating the Simulation Object Model (SOM) Generation Process

Theodore D. Dugone
William S. Murphy Jr.
Leroy A. Jackson
Gerald M. Pearman
U.S. Army TRADOC Analysis Center
PO Box 8692 Monterey, California 93943-0692
(831) 656-4057, 3086, 4061 (DSN 878-x)
dugnonet, murphyw, jacksonl@trac.nps.navy.mil

Keywords:
SOM, FOM, HLA, HLA tools, ADS, Combat Simulation

Abstract: *The United States Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California is conducting research to re-host the Janus combat simulation on a personal computer using the Windows NT operating system. The project will apply advanced technologies to the legacy Janus model including High Level Architecture (HLA) compliance, an innovative modular architecture, an object-oriented design, state-of-art graphical user interfaces, and a modular terrain component. The new simulation will be called HLA Warrior. This paper describes development of an automatic Simulation Object Model (SOM) generation capability being integrated into the new simulation. This feature allows the simulation to automatically self-generate significant portions of a SOM from the application source code, ensuring the published SOM accurately depicts evolving simulation capabilities. SOMs generated from source code eliminate a common HLA federation implementation problem where SOMs are little more than paper specifications. The automatic SOM generation capability capitalizes on techniques that were developed as part of the Analysis Federate project. Analysis Federate research developed the capability to automatically generate the Analysis Federate's SOM from any arbitrary federation's FOM.*

1. Introduction

1.1 Background

The Department of Defense's mandate to transition to the new High Level Architecture (HLA) is causing many verified and validated legacy models that provide a rich set of simulated capabilities and behaviors to be discarded in favor of newer object-oriented HLA compliant replacements. These legacy models are characterized as UNIX based applications designed to operate on vendor specific proprietary hardware. The high cost and enormous overhead associated with procuring, operating, and maintaining these legacy simulations forced the Army to limit their availability to centralized simulation centers. This fielding plan ensured that the simulations were a limited resource and that units had limited access. Computers were not available in troop units during the era in which these legacy simulations were fielded. Subsequent technological advances allowed the Army to issue Windows-based personal computers (PC) to all troop units. Unfortunately, those legacy simulations cannot be used on this prolific hardware forcing troop units to continue to compete for time at a centralized simulation center for access to simulations.

The next generation of Army military simulations such as OneSAF, WARSIM, Combat XXI and AWARS are currently in the systems analysis and design phases. The United States Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California is conducting a re-hosting project to port a legacy simulation from a UNIX-based platform to a PC running Windows NT. This research applies advanced technologies including HLA compliance, a modular architecture, a detailed object-oriented design, state-of-art graphical user interfaces, and a modular terrain component. TRAC-Monterey research will support next generation simulations with insights into hardware platforms and target operating systems.

1.2 HLA Warrior

TRAC-Monterey is re-hosting the high-resolution Janus combat simulation from a Hewlett Packard (HP) workstation onto a PC using the Windows NT operating system. When complete the simulation will be known as

HLA Warrior. The goal of the project is to complete the entire systems analysis, design, and implementation in less than two years to ensure that lessons learned are available early enough to support OneSAF, WARSIM and Combat XXI programs. [1]

TRAC-Monterey is using the HLA Warrior project to develop and demonstrate a new technique to ensure that a simulation's (federate's) Simulation Object Model (SOM) accurately depicts the capabilities that were actually implemented in the federate. In practice, published SOMs often do not reflect the federate's actual implementation because federate developers make programming errors or design changes that are not reflected in the SOM. Discrepancies between the SOM and the actual implementation are often not identified by the HLA Run-Time Infrastructure (RTI) during distributed simulation sessions, causing federates to misinterpret the erroneous objects, attributes, interactions, or parameters.

The new technology that is being introduced in this project allows a simulation to automatically self-generate significant portions of its Simulation Object Model (SOM) from the application source code, ensuring the published SOM accurately depicts the simulation's implemented capabilities. This eliminates the SOM configuration control problem common to HLA federates.

The automatic SOM generation capabilities that are being developed in this research capitalize on techniques that were developed as part of the Analysis Federate project [2]. That research developed the capability of automatically generating the Analysis Federate's SOM from any arbitrary federation's Federation Object Model (FOM), allowing the Analysis Federate to function as a composable component of any federation that uses an object model representation in its FOM [3].

2. HLA Background

The HLA is designed to facilitate simulation interoperability and promote the reuse of simulations as sub-systems or components of larger simulation systems. Object models are the keystones to interoperability and reuse. They describe the capabilities of individual federates and document the information interchanged among federates. The HLA Object Model Template (OMT) establishes the common structure for documenting the capabilities of individual federates in the SOMs and for documenting the exchange of information among federates in the FOMs. It is essential that these object models accurately reflect the capabilities that are implemented. The object models act as the contract among federates. The entire premise of HLA interoperability and reuse is dependent on the accuracy of these object models.

The HLA facilitates simulation interoperability by requiring federations of federates to establish a FOM that identifies the objects, attributes, interactions, and parameters that can be exchanged during a distributed simulation session. The RTI treats the federation's FOM, documented in the Federation Execution Data (FED) file, as a parameter during the initialization of a distributed simulation session. The HLA's treatment of the FOM as a run-time parameter is the mechanism that allows the HLA Analysis Federate to be reused without modification to support federations that use domain specific object models that are not abstractions of a common universal object model. Allowing each federation to establish an object model ensures that the HLA will be capable of providing a distributed simulation interoperability support environment that is flexible enough for existing and future needs.

Simulation reuse in HLA is dependent on the concept that federates must come together and negotiate an object model (FOM) that allows them to satisfy the objectives of a particular federation. Federates are developed to satisfy specific purposes. Each federate has a unique SOM that fully describes its capabilities. A federate's SOM provides a method for advertising those capabilities that the federate can offer to a proposed federation. SOMs are used during the Federation Development and Execution Process (FEDEP) to identify possible candidate federates that meet federation requirements. Federates are chosen for a particular federation because they are known to satisfy certain federation objectives. Therefore, the necessity for the SOM to accurately represent the true implementation of the model is crucial for identifying which federates will best satisfy the federation's objectives. An inaccurate SOM can lead to the selection of a federate that will not meet federation objectives, require re-coding to achieve interoperability, or may cause other interoperability problems.

3. Automatic SOM Generation in Analysis Federate

The Analysis Federate research developed a general-purpose data collection and analysis tool designed to be used as a composable component in any HLA federation that uses an object model representation in its FOM. This required an extension of federate functionality beyond what is called for in the HLA design principles, which promote the use of a federate as a composable component in only those federations that use a common object model abstraction in their FOMs. Without this extension, separate specialized data collection and analysis tools would have to be developed to collect composite data from the distributed simulation sessions for each federation that uses a different object representation in its FOM.

The Analysis Federate research developed a conceptual framework and corresponding fourth generation development tools to provide this composability. These technologies and tools provide the Analysis Federate user with graphical user interfaces to automate the subscription, publication, and interpretation of federation data as well as the ability to automatically generate a unique SOM for each federation that it joins.

The systems design for the Analysis Federate capitalizes on the fact that the OMT formatted FOM contains a robust description of the object, attributes, interactions, and parameters that may be exchanged during a federation execution. The FOM is a paper specification that is used by federate developers to assist them in writing the local RTI component code that allows a federate to communicate with the RTI. The full set of information contained in the FOM is not used online during federation execution. Instead, it is used as a data source by a tool that is used to generate the federation's FED file, which contains a subset of the information in the FOM. The FED is treated as a parameter by the RTI during the initialization of a federation execution. The information contained in the FED is robust enough to allow the RTI to provide the simulation interoperability services for the federates in a federation to exchange data in a consistent manner. However, the FED does not contain enough information to provide data marshalling services for the information that is exchanged during a federation execution. Native data types are among the key categories of information that are contained in the FOM, but are not contained in the FED. Analysis Federate developers realized that if enough information were made available during federation execution to allow a federate to provide its own data marshalling services, then that federate would also have enough information to allow it to develop and implement a systems design that provides for the composability of that federate across federations that use different object model abstractions in their FOMs.

The Analysis Federate obtains this requisite information by treating both the FOM and the FED as parameters during the initialization of a federation. The Analysis Federate's listener component parses the contents of both the FOM and the FED into memory. The information extracted from these files allows the Analysis Federate to automatically take apart and construct the byte streams that are exchanged between a federate and the RTI. It allows the Analysis Federate to automatically apply context to the RTI's byte streams and to function as a composable component of any federation that uses an object model representation in its FOM. The major components of the Analysis Federate are depicted in Figure 1.

One of the fourth generation development tools that is incorporated into the Analysis Federate's Listener component provides a graphical user interface that allows the user to select the object, attributes, and interactions to publish and subscribe. The user selections from the graphical user interface are used to automate the Analysis Federate's subscription and publication services. They are also used to automatically generate the Analysis Federate's SOM.

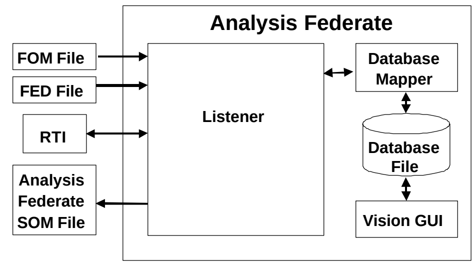

Figure 1. Analysis Federate Systems Design

The Analysis Federate is unique among federates in the manner that it treats its SOM. Most federates use their SOMs for several important purposes. One important use is to assist in the development of the code that is necessary to allow the federate to interact with the RTI and the SOM. Another important use is to assist in the process of forming federations during the FEDEP process. The Analysis Federate does not use its FOM for either of these purposes.

The Analysis Federate uses a federation's FOM as a data source to automatically generate a new SOM for each federation that it joins. Analysis Federate users are not required to write any code to implement the functionality that is documented in its SOM. Instead, the Analysis Federate generates its SOM solely for the purpose of complying with the HLA rules that require each federate to have a SOM. The Analysis Federate does not have any practical use for its SOM beyond any HLA certification testing that may be required for participation in a particular federation.

4. Early SOM Development in HLA Warrior

Janus' FORTRAN procedural language source code was reverse engineered during the systems analysis phase to develop an object-oriented representation of the model's existing functionality. [4,5,6] The object-oriented class design that was developed during this reverse engineering process was documented using Rational Rose diagrams [7]. These diagrams provided code writers with a systems design to guide them through their programming efforts.

SOM developers used the same Rational Rose systems design diagrams for initial SOM development efforts. SOM developers used the Object Model Development Tool (OMDT) to convert Rational Rose diagrams to OMT formatted files that served as the basis for the HLA Warrior SOM. The conversion process significantly reduced the amount of work required to generate initial portions of the SOM and allowed SOM developers to work in parallel with code developers.

The original Rational Rose class diagrams that served as the basis for initial SOM development were intended to accurately reflect the code that would be written. In practice, this was not the case. The class design continued to evolve during the code implementation phase, creating configuration management issues for SOM developers. The code developers modified the class designs to satisfy model efficiency concerns and to ensure HLA Warrior will accurately reflect Janus functionality. These modifications invalidated the diagrams that were used as the basis for the SOM developers' work. It was counterproductive to go through the two-step process of updating the Rational Rose diagrams to reflect implementation changes and subsequently using the OMDT to generate an updated SOM. It was equally counterproductive to interrupt the code development effort to obtain the information needed to manually update the existing SOM. These inefficiencies could be eliminated if the SOM could be automatically generated from the HLA Warrior source code.

5. Automatic SOM Generation in HLA Warrior

The capability to automatically generate a federate's SOM from a federation's FOM was demonstrated in the Analysis Federate. HLA Warrior researchers reasoned that the fourth generation automatic SOM development tools from the Analysis Federate project could be modified to use a federate's source code instead of a FOM as the input stream for generating a SOM. The capability to generate a SOM from source code provided the motivation that led to the development of a system design permitting HLA Warrior to automatically generate its own SOM.

The data elements that describe the objects, attributes, interactions, and parameters within the OMT formatted SOM must exist in a model's source code in order for this automatic SOM generation capability to work. The classes in HLA Warrior are extended to include OMT specific data elements. The techniques developed to extend the HLA Warrior object model are the significant research contributions described in this paper.

HLA Warrior capitalizes on many of the techniques that were developed in Analysis Federate research. The system design for Analysis Federate in Figure 1 is very similar to the HLA Warrior design in Figure 2. Notice that both figures show an automatic SOM generation capability. The difference between the two representations is that Analysis Federate uses the contents of the FOM and FED as the data input source while HLA Warrior uses source code as the data input source.

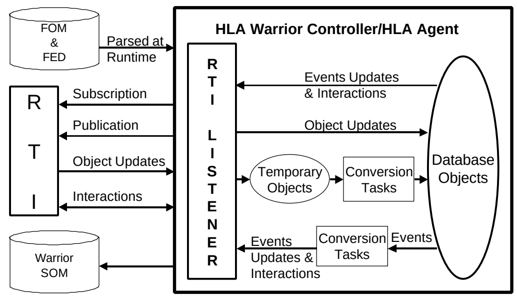

Figure 2. HLA Warrior System Design

The HLA Warrior method of accomplishing automatic SOM generation is to embed the data elements necessary for populating the tables of an OMT formatted SOM directly into the Warrior classes. The OMT specification requires additional information not normally included in coded classes. For example, there are twelve separate fields in the attribute table and eight separate fields in the parameter table for each attribute and parameter within the SOM. The data required to populate some fields within the SOM is not normally related to code development but must still be included in the object classes. Examples of these types of data are accuracy condition and update type.

Automatically generating the object table and attribute table is relatively straightforward. Most information required to populate these tables is already available or easily added to the HLA Warrior object classes. However, generating interaction and parameter tables is more problematic. Simulations written in object-oriented formats implementing a class hierarchy structure do not explicitly identify interactions and parameters. Interactions and parameters may be contained within class methods, but not all methods are interactions. Ongoing research will determine a process to capture interactions and parameters from the simulation code.

The HLA Warrior method of embedding information in the classes is well-suited for the HLA development environment. In other development environments, an alternate approach can be applied. Another mechanism for adding information necessary to complete HLA OMT requirements is to embed comments into the source code that contain the information. If the comments follow a special format, they can be parsed from the source code in a manner similar to code generation. The special parser reads comments and code syntax and produces an interchange file which is used by other tools to produce editable OMT specifications. This is similar to the methodology used with the Java programming language to produce HTML documentation from Java source code and embedded comments.

6. Conclusion

To facilitate rapid, accurate SOM updates that reflect code implementation changes, the HLA Warrior team implemented the concept of automatic SOM generation into HLA Warrior. This feature allows the simulation to automatically self-generate significant portions of a SOM from the application source code. Automated SOM generation ensures the published SOM accurately depicts current code implementation and true capabilities of the model.

This research directly benefits next generation simulation development efforts by providing an automated SOM process. Army After Next (AAN) simulations will be involved in various federations, requiring flexible SOM development techniques. Also, when a simulation is modified or enhanced with additional functionality, the automated process will save SOM developers significant amounts of time updating the object model. Additionally, a SOM derived directly from source code provides a better model representation than current SOM development techniques, supporting verification and validation efforts.

References

1. Jackson, Leroy A., Gerald M. Pearman, and Michael L. McGinnis (1999), "Re-hosting a Constructive Legacy Simulation With Modern Technologies," proceedings of the 1999 Simulation Technology and Training Conference.
2. Jackson, Leroy A. and Ralph J. Wood (1997), "A Federate for Analysis in Advanced Distributed Simulation" 1997 Fall Simulation Interoperability Workshop, September 1997.
3. Murphy, William S., Jr. (1998), "Analysis Federate," Proceedings of the 1998 Fall Simulation Interoperability Workshop, Orlando, FL.
4. Larimer, Larry R., "Building an Object Model of a Legacy Simulation," MS Thesis in Operations Research, Naval Postgraduate School, Monterey, CA, June 1997.
5. Willaims, Julian R. Jr., and Michael J. Saluto (1999), "Re-engineering and Prototyping a Legacy Software System--Janus Version 6.X," MS Thesis in Computer Science, Naval Postgraduate School, Monterey, CA, March 1999.
6. Larimer, Larry R., Arnold Buss, and Leroy Jackson, (1997), "Building a Simulation Object Model of a Legacy Simulation," Proceedings of the 1997 Spring Simulation Interoperability Workshop, Orlando, FL.
7. "Rational Rose 98: Using Rational Rose," Rational Software Corporation, 1998.
8. Jackson, L. A. (1998), "The HLA Warrior Architecture Specification," US Army TRAC-Monterey Technical Report, Monterey, CA.
9. Rieger, Lawrence A., and Gerald M. Pearman, "Re-engineering Legacy Simulations for HLA Compliance", I/ITSEC
10. Knight, Steven D. (1998), "A Comparison of Analysis in DIS and HLA," MS Thesis in Operations Research, Naval Postgraduate School, Monterey, CA, June 1998.
11. Neuberger, T. (1998), "A Progress Report: Recommended Practices for Data Collection in HLA and Other ADS Environments", Paper 129, Proceedings of the Fall 1998 Simulation Interoperability Workshop, Orlando, FL, September 1998.
12. Jackson, Leroy A. and Ralph J. Wood (1997), "A Federate for Analysis in Advanced Distributed Simulation" 1997 Spring Simulation Interoperability Workshop, March 1997.
13. Tapestry Solutions (1998), "Tapestry Kernel Reference Guide."
14. HLA FAQ at http://hla.dmso.mil/
15. Department of Defense: "High-Level Architecture Object Model Template Specification Version 1.3, 5 February 1998.
16. Defense Modeling and Simulation Office: "Federation Development and Execution Process (FEDEP) Model", Version 1.3 Draft, December 9, 1998.
17. Department of Defense Directive 5000.59, "DoD Modeling and Simulation (M&S) Management," January 4,1994.
18. Buss, Arnold H. and Leroy A. Jackson (1998), "Janus Simulation Object Model (J-SOM)," U.S. Army TRAC-Monterey Technical Report No. 98-03, Monterey, CA.

Author Biographies

MAJOR THEODORE D. DUGONE is a U.S. Army officer with over ten years of commissioned service. He graduated with a BS in biology from the Pittsburg State University in 1988 and was commissioned in the Field Artillery. He completed an MS in Operations Research from the Colorado School Of Mines in 1998. He is currently an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California. His current work includes a joint U.S./French federation development project using Warrior.

MAJOR WILLIAM S. MURPHY JR. is a Major in the U.S. Army Corps of Engineers with seventeen years of commissioned service. He graduated with a B.S. from the United States Military Academy in 1982 and with a M.S. in Mathematics and Computer Sciences from the Colorado School of Mines in 1992. Major Murphy is a licensed Professional Engineer (PE) in the Commonwealth of Virginia. He is currently an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California. He is also the Advisor of the Military Operations Research Society's (MORS) Computing Advances in Military Operations Research working group. His current work includes development of the HLA Analysis Federate, a data collection and analysis tool for HLA federations.

MAJOR LEROY A. JACKSON is a U.S. Army officer with over twenty-two years of enlisted and commissioned service. He graduated with a BA in Mathematics from Cameron University in 1990 and with an MS in Operations Research from the Naval Postgraduate School in 1995. Major Jackson is a recipient of the US Army Chief of Staff's Award for Excellence in Operation Research. He is currently an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California and continues graduate studies in operations research at the Naval Postgraduate School.

MAJOR GERALD M. PEARMAN is an Army officer with over twelve years of commissioned service. He graduated from the United States Military Academy at West Point in 1986 and was commissioned in the Aviation branch. He completed an MS in Operations Research from the Naval Postgraduate School in 1997. He is currently an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California where he heads the HLA Warrior project.